C# BASICS

DEFINITIONS:

STRING- blocks of text (numbers are converted into tesxt versions e.g. 1=one)

CONSOLE- looks after the user input and outputs

DOUBLE. – takes whatever follows the statement and converts it into double precision using ‘parsing’ (use of ‘Parse’)

PARSE- (meaning is shown above) - it does this by pulling out each individual digit e.g.’12’ = ten and two digits. This can cause trouble later on, in that if the user doesn’t type in a value or types ‘twenty five’ the Parse method call will not be able to resolve the number and will fail the result

INTEGER- whole numbers e.g. amount of sheep in a field

REALS – decimals/not whole number e.g. temperature

FLOATS – hold DECIMALS

CASTING – the movement from double precision into single

IDENTIFIER- the name a programmer will choose for something within the program

NUMBERS

- Computers manipulate numbers instead of text

- Must tell computer which type of number it is storing and also the range of possible values so we can choose the right type to hold the data as.

- Too much accuracy may slow down the computer, too little accuracy and the wrong values will be given in the end.

**Sbyte- 8 bits: -128 to 127**

**byte -8 bits: 0 to 255**

**short -16 bits: -32768 to 32767**

**ushort- 16 bits: 0 to 65535**

**int -32 bits: - 2147483648 to 2147483647 –STANDARD IN C#**

**uint -32 bits: 0 to 4294967295**

**long- 64 bits: -9223372036854775808 to 9223372036854775807**

**ulong- 64 bits: 0 to 18446744073709551615**

**char -16 bits: 0 to 65535**

Sorting Real Values: (FLOATS)

Range: 1.5E-45 to 1.7E308

Precision of only 7 digits (i.e. less accurate than the pocket calculator). If you want improved accuracy you can use a DOUBLE box, but this will increase the processing time and output will be slower- this precision increases to 15 digits.

DECIMAL TYPE can also be used this holds 27-29 digits but does take double the storage

Storing Real Values:

A float literal can be expressed as a real number with an ‘**f**’ after it:

2.5f

A DOUBLE literal is expressed as a real number WITHOUT the F:

3.5

The compiler is quite fussy about how these numbers can and can’t be combined. This is because when you move from double precision you lose accuracy, so you must take special steps to make it clear that this is what you want to do. This process is known as CASTING.

TEXT

This can be stored as a single character (given by the command ‘CHAR’) or if it’s longer than one character it is stored as an integer.

Strings hold lines of text:

- has to be enclosed within double quotes

- can contain escape sequences

BOOLEAN

A ‘BOOL’ variable is a type of box which can hold whether or not something is true.

Can only output ‘True’ or ‘False’

An example of this would be checking if the network is working. The code would read:

networkOK = true OR false

IDENTIFIERS

Rules:

* All identifiers must start with a letter
* After the letter you can have either letters or numbers or an underscore
* CASE SENSITIVE

**Only use ‘FLOATS’ when absolutely necessary, these will bring inaccuracies into the code which will only later cause more problems. For example, instead of have £1.50 use the value of 150p.**

**Try to be logical in the way that you assign the type to a value. Don’t assign a DOUBLE when the accuracy isn’t needed to be.**

Use of ‘Camel Casing’ (This is the use of capital letters at the start of each new word within one piece of text) this will make it easier to read as you cannot use spaces.

MAKE NAMES MEANINGFUL

GIVING VALUES TO VARIABLES:

C# gets outputs by the use of assignment statements. There are two parts to an assignment, the thing you want to assign and the place you want to put it. E.g:

class Assignment

{

static void Main ()

{

int first, second, third ;

first = 1 ;

second = 2 ;

third = second + first ;

}

}

The first part is just identifying the variables; First, Second and third. These are integers. The last 3 statements are the ones which do the work, these are assignment statements. An assignment gives value to a specified variable, which must be in a suitable type.

The value which is assigned is an expression.

The equals sign – logically- means that you take the result on the right had side of the assignment and drop it into the left, which means that:

2 = second + 1;